

I hereby certify that this paper and/or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 CFR §1.10 on the date indicated below and is addressed to: Mail Stop Patent Application, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Karen Orzechowski

DATE OF DEPOSIT: NOVEMBER 3, 2003

Signature

EXPRESS MAIL LABEL NO.: EV331727366 US

Inventor(s): Hemanta K. Dutta, Seeta Hariharan, Sridhar Rao, and Yanping Wang

## **STATE BASED INGRESS PACKET SELECTION MECHANISM FOR A PACKET PROCESSING SYSTEM IN A NETWORK PROCESSOR**

### **FIELD OF THE INVENTION**

The present invention relates to applications executed by network processors, and more particularly, to the processing of ingress packets by the applications.

### **5 BACKGROUND OF THE INVENTION**

Figure 1 illustrates a conventional network processor system. The system includes a network processor 102 executing an operating system (OS) 108, and one or more state based applications 104 executed by the OS 108. In conventional message processing systems, ingress messages 106 are categorized into classes based on what they accomplish and/or their priority. At any given time, an application 104 may be interested in an ingress message 106, based on the class in which the message 106 belongs and the current state of the application 104.

After the applications 104 are powered on, they eventually reach a state where they

are ready and able to receive operational messages. One conventional approach in processing ingress messages is to use a single message queue for all messages. As each message is dequeued, the application 104 “wakes”, after which it determines if it is interested in the message 106 from the queue. If so, then the message 106 is processed by the application 104. If not, the message 106 is either dropped or deferred by putting it back in the queue. However, with this approach, the application 104 wakes each time a message 106 is received, even if the application 104 is not interested in the message. This approach is thus inefficient and results in the slow execution of the application 104. In addition, not all network processors have the defer capability. Even when it is available, messages of the same class can be processed out of order. A single message queue also may not be able to supply the number of priority levels needed by the application 104.

Another conventional approach is to use one message queue per class and for the application 104 to wait on the appropriate queue. This approach is efficient only if the application 104 is interested in just one message class and blocks on the corresponding queue. However, when the application 104 wishes to receive more than one class of messages, the application 104 will only see class of messages from the queue at which it is blocking, and will not see any of the messages from the other classes of messages at other queues. One technique is to poll for available messages. At regular intervals, the application 104 checks for a message in each queue in which it is interested. However, this approach has an inherent latency since pending messages will not be detected until the next poll cycle. In addition, polling creates overhead for the network processor 102, even when there are no messages in the interested queues. This approach is thus inefficient and results

in the slow execution of the application 104.

Accordingly, there exists a need for a more efficient method for performing state based ingress packet selection for a packet processing system in a network processor. The present invention addresses such a need.

5

## **SUMMARY OF THE INVENTION**

An efficient method for performing state based ingress packet selection for a packet processing system in a network processor is disclosed. With this method, each set of message classes is associated with a state of an application, and each set of message classes is assigned to a semaphore. The application blocks one of the semaphores based on its current state. When a message is received, and the message belongs to a class assigned to the blocked semaphore, the blocked semaphore is signaled. The blocked semaphore then wakes the application. In this manner, when the application is awakened, it is guaranteed to find a message that it is interested in processing in its current state, resulting in greater efficiency in the processing of ingress messages.

10

15

## **BRIEF DESCRIPTION OF THE FIGURES**

Figure 1 illustrates a conventional network processor system.

Figure 2 is a flowchart illustrating a preferred embodiment of a method for performing state based ingress packet selection for a packet processing system in a network processor in accordance with the present invention.

20

Figure 3 illustrates a preferred embodiment of a network processor system utilizing

the method in accordance with the present invention.

Figure 4 is a flowchart illustrating in more detail the method for performing state based ingress packet selection for a packet processing system in a network processor in accordance with the present invention.

5

## **DETAILED DESCRIPTION**

The present invention provides a more efficient method for performing state based ingress packet selection for a packet processing system in a network processor. The following description is presented to enable one of ordinary skill in the art to make and use the invention and is provided in the context of a patent application and its requirements. Various modifications to the preferred embodiment will be readily apparent to those skilled in the art and the generic principles herein may be applied to other embodiments. Thus, the present invention is not intended to be limited to the embodiment shown but is to be accorded the widest scope consistent with the principles and features described herein.

10

15

To more particularly describe the features of the present invention, please refer to Figures 2 through 4 in conjunction with the discussion below.

Figure 2 is a flowchart illustrating a preferred embodiment of a method for performing state based ingress packet selection for a packet processing system in a network processor in accordance with the present invention. Figure 3 illustrates a preferred embodiment of a network processor system utilizing the method in accordance with the present invention. Referring to both Figures 2 and 3, first, a set of message classes are assigned to a semaphore 310, via step 202, where the set of message classes is associated

20

with a state of an application 304. The set of message classes and the set of application states are defined at application design time. Associating a set of message classes with a state of an application means that the application is interested in ingress messages that belong to the set of message classes while it is in that state. A semaphore is a basic facility of the operating system (OS) 308. It is used to wake applications and are known in the art. The present invention utilizes a semaphore 310 by assigning a set of message classes to it. When a message 306 is received, via step 204, the OS 308 determines if the message 306 belongs to a message class in the set of message classes, via step 206. If so, then the semaphore 310 assigned the set of message classes wakes the application 304. Since the set of message classes are assigned to the semaphore 310, and since the set of classes are associated with a state of the application interested in messages belonging to the set of classes, the semaphore 310 only wakes the application 304 when a message in which it is interested is received.

Figure 4 is a flowchart illustrating in more detail the method for performing state based ingress packet selection for a packet processing system in a network processor in accordance with the present invention. Assume that a set of message classes (c1, c2, etc.) and a set of application states (s1, s2, etc.) are defined at application design time. A mapping is established from each application state to a set of message classes that can be received while in that state. The following is an example mapping:

s1 -> (c1, c2)  
s2 -> (c3)  
s3 -> (c2, c3, c4)  
s4 -> (c1, c2)

In this example, when an application is in state s1, it is interested in messages belonging to classes c1 and c2; when in state s2, it is interested in messages belonging to class c3; when in state s3, it is interested in messages belonging to classes c2, c3, and c4; and when in state s4, it is interested in messages belonging to classes c1, and c2.

5           According to the present invention, each set of message classes (c1, c2, etc.) is assigned to one of a plurality of semaphores 310-314, via step 402, where each set of message classes is associated with one of a plurality of states of the application 304. For the above example, the following assignments are made:

10           (c1, c2) -> SEM 310  
            (c3) -> SEM 312  
            (c2, c3, c4) -> SEM 314

15           The set of message classes containing c1 and c2 is assigned to semaphore 310; the set of message classes containing c3 is assigned to semaphore 312; and the set of message classes containing c2, c3, and c4 is assigned to semaphore 314.

            Next, the application 304 blocks one of the plurality of semaphores 310-314 based on its current state, via step 404. For example, if the application 304 is currently in s1, it blocks semaphore 310; if the application 304 is currently in s2, it blocks semaphore 312; and if the application 304 is currently in s3, it blocks semaphore 306.

20           When the OS 308 receives a message 306, via step 406, it determines the message class to which the message 306 belongs, via step 408. The OS 308 then signals the blocked semaphore, if the message belongs to the set of message classes assigned to the blocked semaphore, via step 410. The signaled semaphore then wakes the application 304, via step 412.

For example, if the application 304 blocked semaphore 310, and the message 306 belongs to c1 or c2, then semaphore 310 is signaled, via step 410, since the set of message classes containing c1 and c2 was assigned to semaphore 310. Semaphore 310 then wakes the application 304, via step 412. If the application 304 blocked semaphore 312, and the message 306 belongs to c3, then semaphore 312 is signaled, via step 410, since the set of message classes containing c3 was assigned to semaphore 312. Semaphore 312 then wakes the application 304, via step 412. If the application 304 blocked semaphore 314, and the message 306 belongs to c2, c3, or c4, then semaphore 314 is signaled, via step 410, since the set of message classes containing c2, c3, and c4 was assigned to semaphore 314. Semaphore 314 then wakes the application 304, via step 412. Once awake, the application 304 processes the message 306, via step 414.

In this manner, when the application is awakened, it is guaranteed to find a message that it is interested in processing in its current state. If there are no messages, none of the semaphores are signaled, and the application continues to block. Other applications can be scheduled on the processor. Thus, with the present invention, no polling is needed to check for messages when there are no messages.

When a message is received but the application is not interested in that message, i.e., the message does not belong to a class assigned to the blocked semaphore, the message can either be dropped or deposited to a buffer to be processed if the application enters the state that is interested in the message.

A more efficient method for performing state based ingress packet selection for a packet processing system in a network processor has been disclosed. With this method, each

set of message classes is associated with a state of an application, and each set of message classes is assigned to a semaphore. The application blocks one of the semaphores based on its current state. When a message is received, and the message belongs to a class assigned to the blocked semaphore, the blocked semaphore is signaled. The blocked semaphore then  
5 wakes the application. In this manner, when the application is awakened, it is guaranteed to find a message that it is interested in processing in its current state, resulting in greater efficiency in the processing of ingress messages.

Although the present invention has been described in accordance with the embodiments shown, one of ordinary skill in the art will readily recognize that there could  
10 be variations to the embodiments and those variations would be within the spirit and scope of the present invention. Accordingly, many modifications may be made by one of ordinary skill in the art without departing from the spirit and scope of the appended claims.